Università Telematica Internazionale Uninettuno

Facoltà di Ingegneria

Corso di Laurea Magistrale in Ingegneria Informatica indirizzo Big Data

Introduzione ai Big Data

Anno Accademico 2017/2018

Allegato tecnico Progetto1

Roberto Favaroni
matricola 199HHHINGINFOR

Pseudocodice MapReduce

Job1

```
#key=non significativo
#value=blocco di linee del file di input
method Map(key, value)
        for all line E value do
                EMIT((month, product), 1)

#key=mese,prodotto
#value=numero occorrenze
method Reduce(key, values)
        sum <- 0
        for all occurrences E values do
                sum <- sum + occurrences
        EMIT(key,sum)
```

Job2

```
#key=non significativo
#value=blocco di linee del file di input
method Map(key, value)
        for all line E value do
                EMIT((month, product), 1)

#key=mese,prodotto
#value=numero occorrenze
method Reduce(key, values)
        product <- key[2]
        price <- price in pricesTable by product
        for all occurrences E values do
                sum <- sum + occurrences
        EMIT(key,sum*price)
```

Job3

```
#key=non significativo
#value=blocco di linee del file di input
method Map(key, value)
        for all line E value do
                products <- extract from line
                combinations <- calculate_pairs_combinations (products, 2)
                for all combination E combinations do
```

```
                append item pair combination[1] and combination[2] to keys
            append rowcount-item to keys
            for all product in products do
                append product-item to keys
        EMIT(keys 1)

#key=coppia prodotti | singolo prodotto | conteggio scontrini
#value=numero occorrenze
method Reduce(key, values)
        sum <- 0
        for all occurrences E values do
            sum <- sum + occurrences
        EMIT(key,sum)
```

Pseudocodice R

Job1

MapReduce
init Hadoop MR environment
set application constants
remove if present last output
inputVar <- read input file from HDFS
results <**- call** MapReduce(inputVar, outputVar, map function, reduce function)
set output results file

no MapReduce (R only)
open input file
    for all line E file **do**
        fields <- split(line)
        currMonth <- fileds[0]
        for all field[2:max] E line **do**
            outList[field]++
close input file

pseudocodice comune elaborazione data frame per risultato finale
#per ciascun mese del 2015, i cinque prodotti piu venduti seguiti dal numero complessivo di pezzi venduti
#2015-01: pane 852, latte 753, carne 544, vino 501, pesce 488
for all month in year **do**
    currMonth <- results of month
    topFive <- extract first 5 from currMonth order by occurrences descending
    write topFive into output results file

Job2

MapReduce
init Hadoop MR environment
set application constants
remove if present last output
pricesTable <- read local input file
inputVar <- read input file from HDFS
results <**- call** MapReduce(inputVar, outputVar, map function, reduce function)

no MapReduce (R only)
open input file
    for all line E file **do**
        fields <- split(line)
        currMonth <- fileds[0]
        for all field[2:max] E line **do**
            prodPrice <- extract from productList
            outList[field]+=prodPrice
close input file

pseudocodice comune elaborazione data frame per risultato finale
#per ciascun prodotto, l'incasso totale per quel prodotto di ciascun mese del 2015
#pane 1/2015:12340 2/2015:8530 3/2015:9450 …
resultsByProduct <- extract from results order by product, month
for all product in resultsByProduct **do**
    write product, revenue into output results file

Job3

MapReduce
init Hadoop MR environment
set application constants
remove if present last output
inputVar <- read input file from HDFS
results <**- call** MapReduce(inputVar, outputVar, map function, reduce function)

no MapReduce (R only)
open input file
    for all line E file **do**
        products <- extract from line
        combinations <- calculate_pairs_combinations (products, 2)
        for all combination E combinations **do**
            outList[combination[1], combination[2]]++
        outList[“rowcount”]]++
        for all product in products do
            outList[product]++
close input file

pseudocodice comune elaborazione data frame per risultato finale
#per ciascuna coppia di prodotti (p1,p2):
#(i) la percentuale del numero complessivo di scontrini nei quali i due prodotti compaiono insieme (supporto della regola di associazione p1->p2)
#(ii) la percentuale del numero di scontrini che contengono p1 nei quali compare anche p2 (confidenza della regola di associazione p1->p2)
#pane,latte,30%, 4%

rowCountResult <- extract from results by key=rowcount
productsPairResult <- extract from results by key=productsPair
productsResult.df <- extract from results by key=singleProduct

#per ogni coppia di prodotti calcolo supporto e confidenza
for all productPair E productsPairResult **do**
    productPairLabel <- productPair[1]
    productPairCount <- productPair[2]
    support <- productPairCount / rowCountResult * 100
    confidence <- productPairCount / (extract from productsResult by productPairLabel[1]) * 100
    write support, confidence, into output results file

Risultati test scalabilità

input size class:

- mediumInput: file da 100K scontrini 2.5 MB
- largeInput: file da 400K scontrini 10 MB

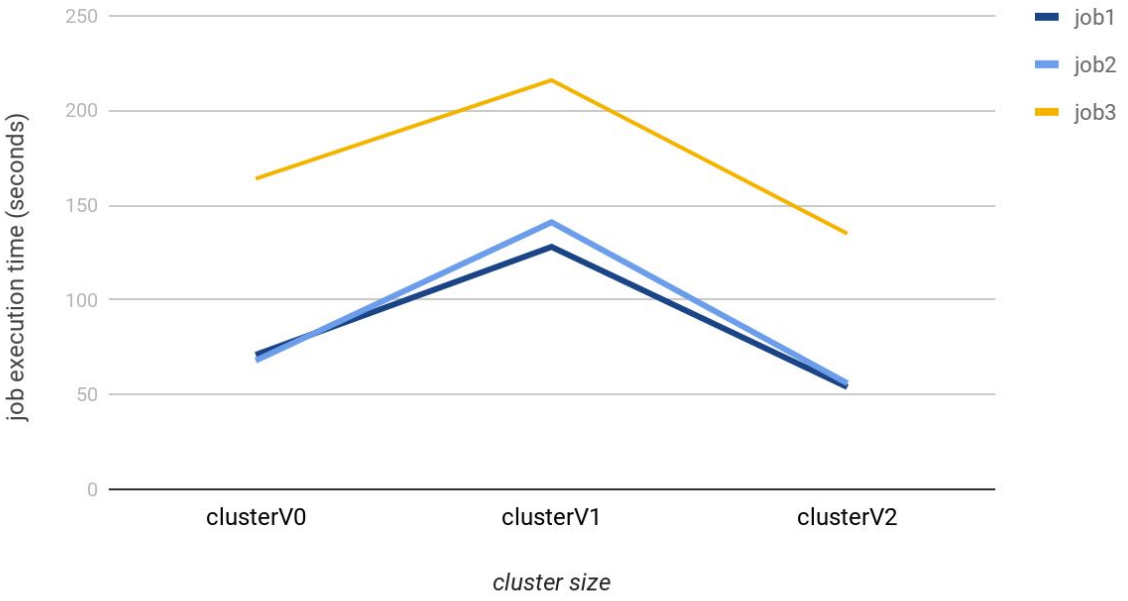
cluster class:

- clusterV0: 1 nodo VM VirtualBox, 4 CPU
- clusterV1: 2 nodi VMs VirtualBox, 2 CPU per nodo
- clusterV2: 2 nodi VMs VirtualBox, 4 CPU per nodo

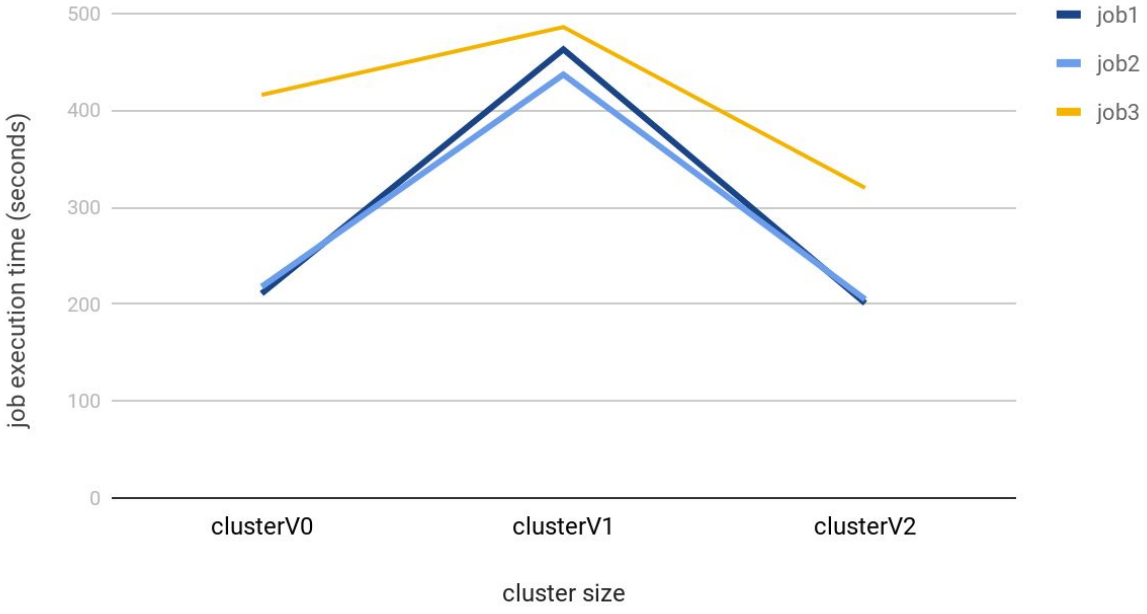
input size class	job	clusterV0 job execution time (seconds)	clusterV1 job execution time (seconds)	clusterV2 job execution time (seconds)
mediumInput	job1	71	128	54
	job2	68	141	56
	job3	164	216	135
largeInput	job1	211	463	201
	job2	218	437	205
	job3	416	486	320

Nota: i tempi di esecuzione risentono fortemente del fatto che i test sono stati eseguiti utilizzando macchine virtuali ospitati sullo stesso host; scalabilità più efficace si sarebbe evidenziata operando su nodi virtuali mediante infrastruttura cloud IaaS o avvalendosi di nodi fisici differenti.

Scalability chart - mediumInput



Scalability chart - largeInput



Info per esecuzione test

Per tutti i job l'input è rappresentato dal file *scontrini.txt* contenente 20 righe/scontrini.

Prima dell'esecuzione occorre:

- impostare le seguenti variabili d'ambiente in funzione delle proprie installazioni di Hadoop e JRE:
  export JAVA_HOME=[java_home]
  export HADOOP_HOME=[hadoop_home]
  export HADOOP_STREAMING=[hadoop_streaming_jar]
  export PATH=$PATH:$HADOOP_HOME/bin
- installare la libreria R *lubridate* e tutte le librerie e dipendenze necessarie per l'esecuzione di codice RHadoop MapReduce (come da slide videolezione)
- avviare correttamente ambiente Hadoop HDFS e YARN
- accedere alla directory decompressa contenente i file sorgenti e i file *scontrini.txt* e *prodotti.txt*
- caricare il file di input *scontrini.txt* su HDFS nella directory da creare */progetto1*:
  >hdfs dfs -mkdir /progetto1
  >hadoop dfs -put scontrini.txt /progetto1
  Il file *prodotti.txt* va invece mantenuto in locale
- lanciare i singoli job da linea di comando:
  >Rscript ./job[x].R per job R/MapReduce
  >Rscript ./job[x]-noMR.R per job solo R
- per visualizzare i risultati accedere al rispettivo file di output locale (cfr. infra) presente nella directory corrente

Job1

esecuzione:
Rscript job1.R
output file:
 ./top5PerMese.txt
formato output file:
[mese1/anno]: [prodotto1] [numVendite], [prodotto2] [numVendite], … [prodottoN] [numVendite]
esempio:
2015-01: vino 2, miele 2, marmellata 2, biscotti 1, latte 1

Job2

esecuzione:
Rscript job2.R
output file:
./incassoMesePerProdotto.txt
formato output file:
[prodotto] [mese1/anno]: [incasso] [mese2/anno]: [incasso] … [meseN/anno]: [incasso]
esempio:
acqua 2/2015:0.4 3/2015:0.4 7/2015:0.4 9/2015:0.8 10/2015:0.4

Job3

esecuzione:
Rscript job3.R
output file:
./regoleAssociazione.txt
formato output file:
[prodotto1], [prodotto2], [%supporto], [%confidenza]
esempio:
olio, latte, %15.00, %50.00

File allegati

- job1.R — file sorgente R/MapReduce job1
- job2.R — file sorgente R/MapReduce job2
- job3.R — file sorgente R/MapReduce job3
- job1-noMR.R — file sorgente R job1
- job2-noMR.R — file sorgente R job2
- job3-noMR.R — file sorgente R job3
- scontrini.txt — file di input con 20 scontrini
- prodotti.txt — file con associazione prodotto->prezzo
- top5PerMese.txt — file di output per job1 con input 20 scontrini
- incassoMesePerProdotto.txt — file di output per job2 con input 20 scontrini
- regoleAssociazione.txt — file di output per job3 con input 20 scontrini
- top5PerMese-noMR.txt — file di output per job1-noMR con input 20 scontrini
- incassoMesePerProdotto-noMR.txt — file di output per job2-noMR con input 20 scontrini
- regoleAssociazione-noMR.txt — file di output per job3-noMR con input 20 scontrini